ГЛАВА XX.
Решающие деревья

Удивительно, что мы видим деревья
и больше не удивляемся им
Р.У. Эмерсон

Программирование заставило дерево зацвести
А. Перлис

Решающее дерево (Decision Tree) приведено на рис. XX.1, это дерево[1], листьям[2] (leaf) которого сопоставлены метки. Внутренним вершинам (internal nodes) – **предикаты**, а дугам – значения предикатов. В задаче классификации каждому листу может сопоставляться набор меток с вероятностями. Говорят, что во внутренних вершинах «**производится ветвление / расщепление по предикату**». Также предполагается, что в дереве выделена одна из вершин, которая называется **корнем**, дерево ориентировано, из каждой внутренней вершины исходит две дуги (вершины, в которые они ведут, называются **потомками** данной), в каждую вершину, кроме корня, заходит одна дуга (на рис. XX.1 направления дуг не указаны). Процесс определения метки по объекту выглядит следующим образом: стартуем с корня, проверяем значение предиката и переходим по соответствующей дуге в одного из потомков, процесс продолжается до тех пор, пока не попадаем в лист с меткой, которая и является ответом. Приведём пример с деревом на рис. XX.1 и объектом – описанием клиента со значениями признаков «доход» = 55000, «число просрочек» = 2. Проверяем значение предиката в корне, поскольку доход клиента больше порога 30000, переходим в правое поддерево, а поскольку число просрочек у клиента также больше порога 1, переходим в лист с меткой «не выдавать» (видимо, имеется в виду кредит).

Часто используют аббревиатуру **CART (Classification and Regression Trees)** для обозначения решающих деревьев, но, на самом деле, это лишь один из стандартных подходов к их построению.

[1] Дерево – связный граф без циклов.
[2] Листья также называют терминальными вершинами (terminal nodes).

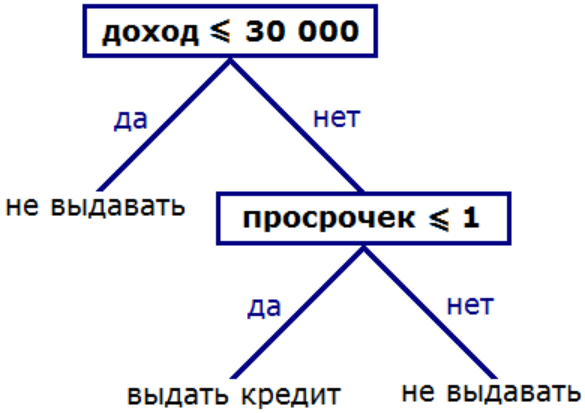

Рис. XX.1. Решающее дерево в задаче классификации.

В этой главе мы рассматриваем **бинарные деревья (binary trees)**, в которых каждая вершина имеет двух потомков (а в общем случае их может быть больше). Перечислим наиболее популярные предикаты. Для вещественного признака чаще используют **сравнение признака с порогом**

$$P(x\,|\,i,\theta) = I[\,f_i(x) \le \theta] \qquad \text{(XX.1)}$$

($f_i(x)$ – значение i-го признака объекта x, $\theta \in \mathbb{R}$ – порог для сравнения), а для категориального –

$$P(x\,|\,i,C) = I[\,f_i(x) \in C], \qquad \text{(XX.2)}$$

где C – подмножество категорий i-го признака. Но возможны и «более экзотические предикаты», например в т.н. **«косых деревьях» (oblique decision trees / BSP: binary space partition trees)** используется предикат сравнения линейной комбинации признаков с порогом

$$P(x\,|\,\{w_i\}_{i=1}^n, \theta) = I\left[\sum_{i=1}^n w_i f_i(x) \le \theta\right],$$

а в **«сферических» (sphere trees)** – проверка на нахождение внутри шара

$$P(x\,|\,\{z_i\}_{i=1}^n, \theta) = I\left[\sum_{i=1}^n (f_i(x) - z_i)^2\right] \le \theta^2.$$

Теоретически предикат может быть любым, но на практике возникает проблема эффективного построения оптимального или почти-оптимального дерева для выбранного вида предиката и функционала качества, поэтому предикаты не выбирают «слишком сложными», чаще предпочитают (XX.1) и (XX.2).

Расщеплению по переменной (splitting) соответствует разбиение (stratifying / segmenting) пространства объектов на области (регионы). На рис. XX.2 показано, что если дерево ограничить одним уровнем, то при использовании предиката (XX.1) оно делит пространство на два полупространства. Кстати, такие одноуровневые решающие деревья называют **решающими пнями (decisive stumps)**.

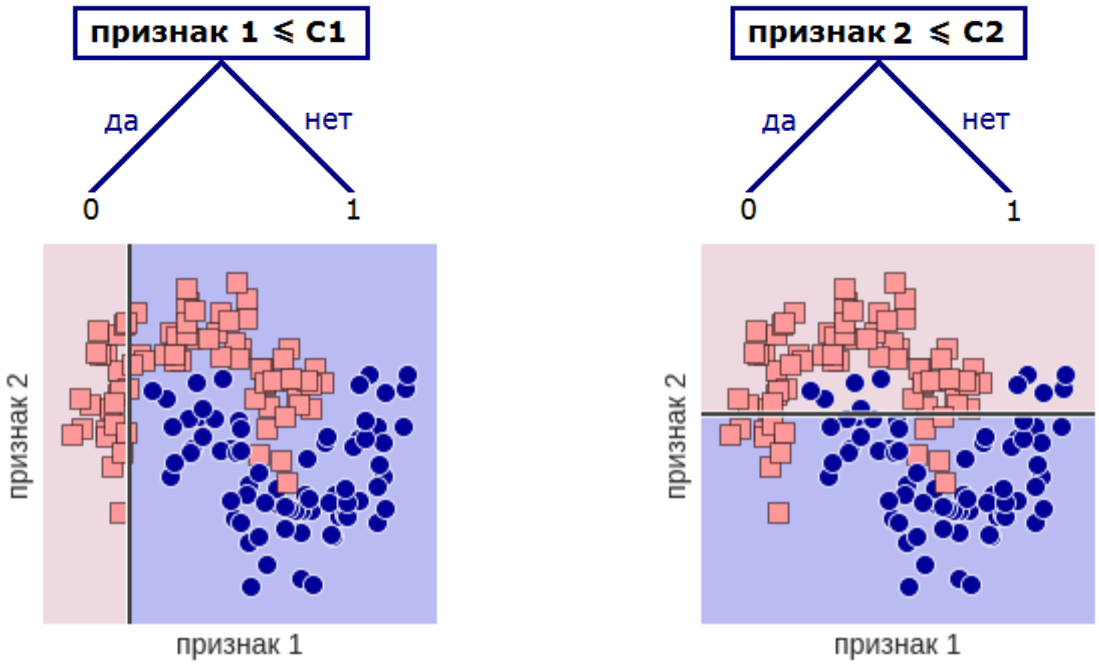

Рис. XX.2. Разбиение на области решающими пнями.

Более сложное дерево всё равно имеет довольно простую решающую поверхность – кусочно-постоянную:

$$a(x) = \sum_j a_{R_j} I[x \in R_j], \qquad \text{(XX.3)}$$

суммирование производится по всем непересекающимся областям (точнее их номерам), на которое разбивается признаковое пространство $\aleph$ деревом:

$$\bigcup_j R_j = \aleph,$$

$$R_i \cap R_j = \varnothing \quad \forall i \neq j.$$

Визуально решающая поверхность может быть даже проще, чем это ожидается из приведённых формул, поскольку, например, в задаче классификации в соседних регионах могут быть одинаковые метки. На рис. XX.3 мы видим, что с точки зрения расстановки меток признаковое пространство разбилось на 2 связные области. Соседние области с одинаковыми метками визуально

сливаются в одну. Кроме того, формально полученное дерево можно упростить, поскольку в левом поддереве метки у всех листьев совпадают и сравнение «признак 1 ≤ C1» можно сразу заменить на метку 1.

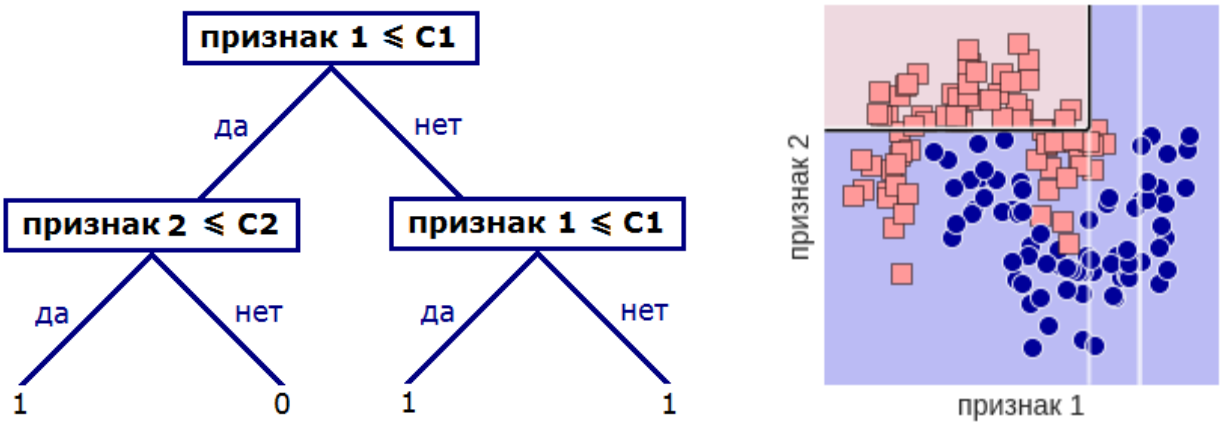

Рис. XX.3. Решающая поверхность дерева

Аналогично, решающие деревья можно строить в задаче регрессии, только метки тут будут вещественными числами. На рис. XX.4-5 показана регрессия с помощью решающих деревьев разной глубины. Пока мы не говорили, как строятся по обучающей выборке решающие деревья, но, как и в задаче классификации на рис. XX.3, решающее дерево «огрубляет информацию из обучения». Увеличение глубины приводит к «затачиванию» на выборку, т.к. листья становятся всё меньше (по числу попавших в них объектов обучения) и ошибка дерева на обучении уменьшается.

Впрочем, любая модель сжато описывает обучающую выборку.

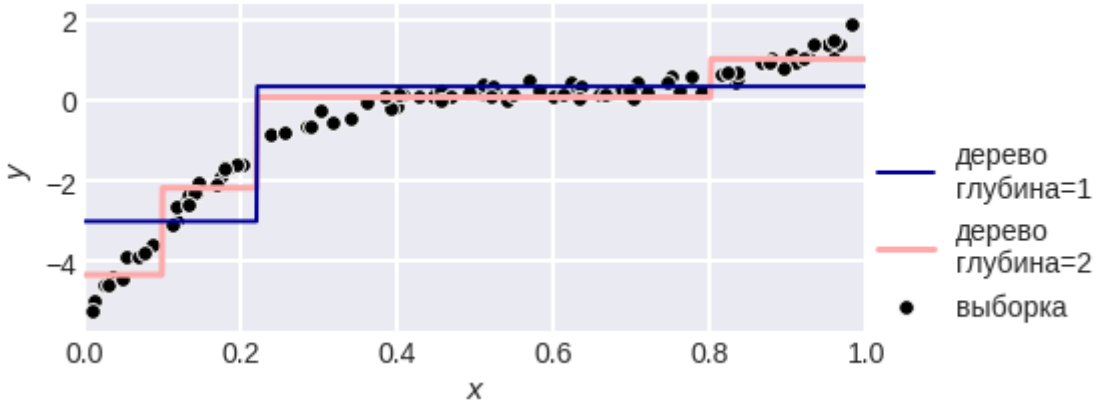

Рис. XX.4. Регрессия с помощью решающих деревьев.

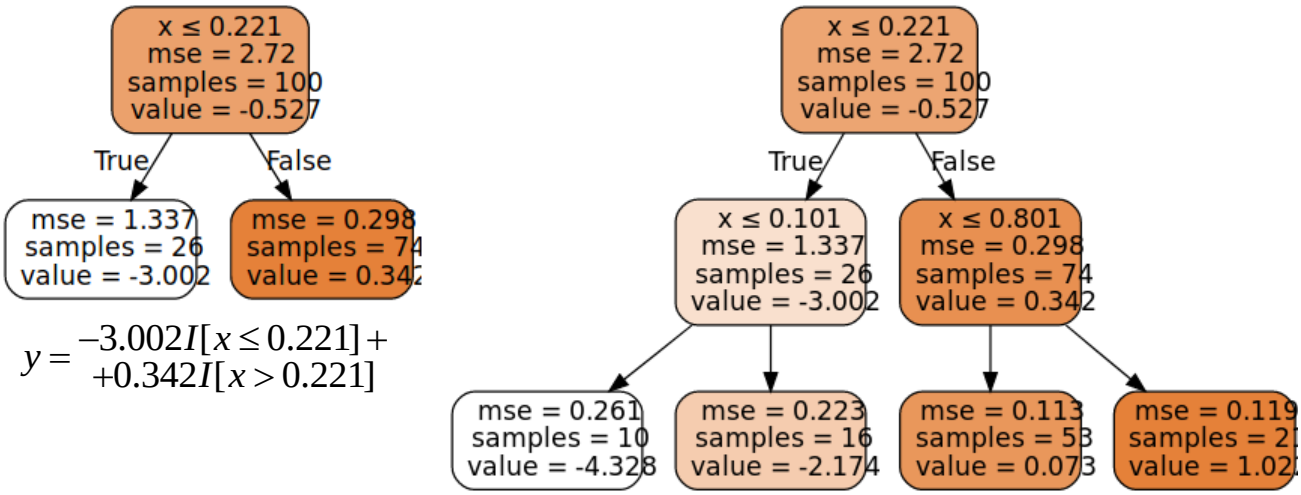

Рис. XX.5. Примеры решающих деревьев в задаче регрессии.

Идея построения дерева и определение меток в листьях

Рассмотрим теперь, как осуществляется **построение дерева** на примере задачи регрессии. В идеале в задаче регрессии с функцией ошибки MSE[1] нужно решить такую задачу минимизации:

$$\sum_i \sum_j I[x_i \in R_j](y_i - a_{R_j})^2 \to \min,$$

где минимизация проводится по всем разбиениям на области $\{R_j\}$, реализуемым с помощью дерева, и по всем выборам «меток листьев» a_{R_j}. Само выражение для минимизации называется **Residual Sum of Squares (RSS)** – сумма квадратов невязок (знакома нам по линейной регрессии). Указанная минимизация довольно трудоёмка[2], поэтому на практике строят дерево «по уровням», последовательно минимизируя RSS – **жадный подход сверху-вниз (top-down greedy approach)**. Стартуя от дерева, состоящего из одной вершины, можно проводить расщепления выбирая признак и порог так, чтобы минимизировать значение RSS. Расщепления производятся пока не выполнятся некоторые критерии останова (ограничения на глубину дерева, число объектов обучающей выборки в листьях, на изменение RSS – об этом подробнее дальше).

Заметим, что если зафиксировать дерево, то оптимальные метки в листьях очевидны. Они соответствуют оптимальным константным решениям по

[1] См. главу «Функции ошибки».

[2] В некоторых постановках построение оптимального дерева является NP-полной задачей.

областям, например, в задаче регрессии с функцией ошибки MSE разумно усреднить метки объектов обучающей выборки, которые попалив эту область:

$$a_{R_j} = \frac{1}{|\{x_i : x_i \in R_j\}|} \sum_{x_i \in R_j} y_i \,.$$

Поэтому в построенном дереве листьям приписываются метки по очень простым правилам: в задаче регрессии как показано выше, а в задаче классификации чаще приписывают самую популярную метку среди объектов обучающей выборки, которые попалив эту область:

$$a_{R_j} = \text{mode}(\{y_i : x_i \in R_j\})$$

(это, кстати, оптимальное константное решение для задачи классификации с функцией ошибки accuracy – доля верных ответов). Хотя возможны и другие стратегии приписываний меток, заточенные под специальные функционалы качества.

Построение решающего дерева в задаче классификации

В задаче классификации построение дерева также осуществляют согласно описанному выше жадному подходу сверху-вниз (top-down greedy approach). Основная тонкость при построении – что оптимизировать при выборе признака и порога для расщепления. Например, на рис. XX.6 изображены значения признака и оценки распределений этих значений у объектов разных классов, какой порог выбрать?

Рис. XX.6. Пример распределений значений признака у объектов разных классов.

Интуитивно, надо выбирать порог так, чтобы в листьях дерева объекты были одного или почти одного класса. На рис. XX.7 показаны хорошее и плохое расщепления признака при построении решающего дерева.

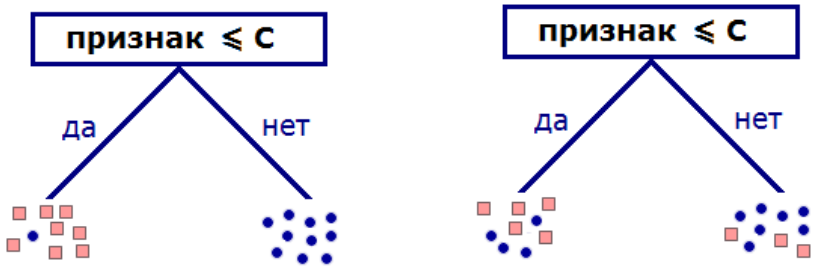

Рис. XX.7. Хорошее (слева) и плохое (справа) расщепление.

Основная идея выбора порога θ на признаке f: ввести **меру неоднородности / зашумлённости (impurity)** множества $H(R)$, которая оценивает, насколько в области «почти все объекты одного класса», тогда при расщеплении области R на две подобласти $R_{\text{left}} = \{x \in R \mid f(x) \le \theta\}$ и $R_{\text{right}} = \{x \in R \mid f(x) > \theta\}$ можно оптимизировать весовое усреднение мер неоднородностей:

$$\frac{|R_{\text{left}}|}{|R|} H(R_{\text{left}}) + \frac{|R_{\text{right}}|}{|R|} H(R_{\text{right}}),$$

где $|R|$ – число объектов обучения в области R, аналогично с R_{left} и R_{right}. Обычно смотрят на сколько изменится неоднородность, поэтому из меры неоднородности области R (до расщепления) вычитают весовое усреднение мер неоднородностей областей после расщепления:

$$Q(R,\theta) = H(R) - \frac{|R_{\text{left}}|}{|R|} H(R_{\text{left}}) - \frac{|R_{\text{right}}|}{|R|} H(R_{\text{right}}), \qquad \text{(XX.4)}$$

Веса пропорциональные мощностям областей после расщепления нужны для того, чтобы улучшить сбалансированность дерева: формально выгодно расщеплять, откусывая маленькие кусочки выборки – несколько объектов одного класса попадают в левую или правую область. Потом мы ещё вернёмся к теоретическому обоснованию этой формулы. На рис. XX.8 показаны значения одной популярной меры неоднородности и функционала $Q(R,\theta)$ в зависимости от выбора порога θ. Заметим, что весовые множители делают невыгодным несбалансированные разбиения области (когда подобласти существенно различаются по мощности).

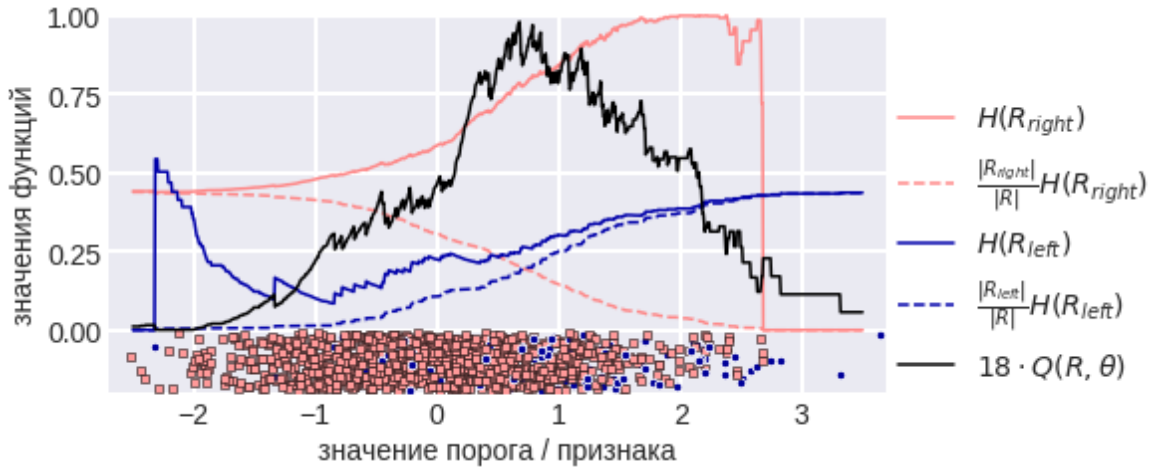

Рис. XX.8. Меры неоднородностей в листьях при разных порогах и $Q(R,\theta)$.

Приведём примеры классических мер неоднородности в задачах классификации. Пусть есть область R, в которой доли объектов всех классов: $p_1,\ldots,p_l$. Используются следующие меры неоднородности:

Название критерия расщепления	Общая формула для $H(R)$	Частный случай для двух классов, $p_1 = p$, $p_2 = 1-p$
неправильной классификации (Missclassification criteria)	$1 - p_{\max}$	$\min[p, 1-p]$
Энтропийный	$-\sum_j p_j \log_2 p_j$	$-p\log_2 p - (1-p)\log_2(1-p)$
Джини	$\sum_j p_j(1-p_j) = 1 - \sum_j p_j^2$	$2p(1-p) = 1 - p^2 - (1-p)^2$

Заметим, что любая из приведённых функций обращается в ноль (и это её минимальное значение) только если все объекты принадлежат одному классу (считаем, что $0\log_2 0 = 0$) и достигает своего максимального значения при

$$p_1 = \ldots = p_l = \frac{1}{l}$$

(когда доли объектов всех классов равны). На рис. XX.9 показаны графики мер неоднородностей в частном случае двух классов от доли объектов первого

класса $p_1 = p$, поскольку доля объектов второго в этом случае выражается по формуле $p_2 = 1 - p$. При этом значения джини и МС увеличены в два раза, чтобы максимумы всех функций совпадали. Поведение всех функций очень логичное: при возрастании доли объектов класса 1 от 0 до 0.5 они возрастают, при дальнейшем возрастании доли – убывают, графики симметричны относительно прямой $p = 0.5$.

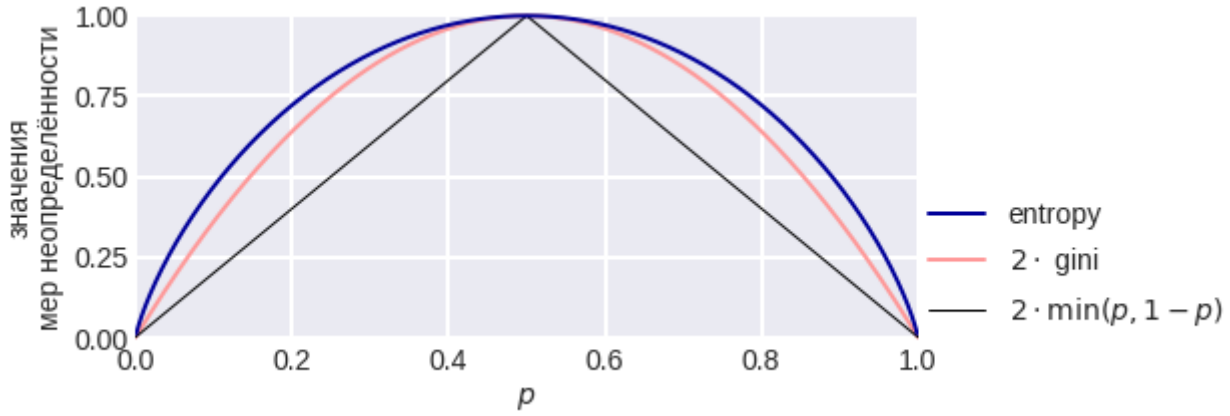

Рис. XX.9. Меры неопределённости в случае бинарной классификации.

Есть ещё также менее распространённые критерии расщепления, не записывающиеся через меры неоднородности:

$$Q(R,\theta) = \left| \frac{|R_{\text{right}} \cap K_0|}{|K_0|} - \frac{|R_{\text{right}} \cap K_1|}{|K_1|} \right| =$$

$$= \left| \frac{|R_{\text{left}} \cap K_0|}{|K_0|} - \frac{|R_{\text{left}} \cap K_1|}{|K_1|} \right|$$

– его мы выведем при рассмотрении функционала качества AUC_ROC, и **Twoing** –

$$Q(R,\theta) = \frac{1}{4} \frac{|R_{\text{left}}|}{|R|} \frac{|R_{\text{right}}|}{|R|} \left(\sum_{j=1}^{l} \left| p_j(R_{\text{left}}) - p_j(R_{\text{right}}) \right| \right)^2.$$

Ни рис. XX.10-11 показаны графики оптимизируемых функций при использовании разных критериев расщепления. Максимум каждого графика соответствует оптимальной точке расщепления с соответствующим критерием.

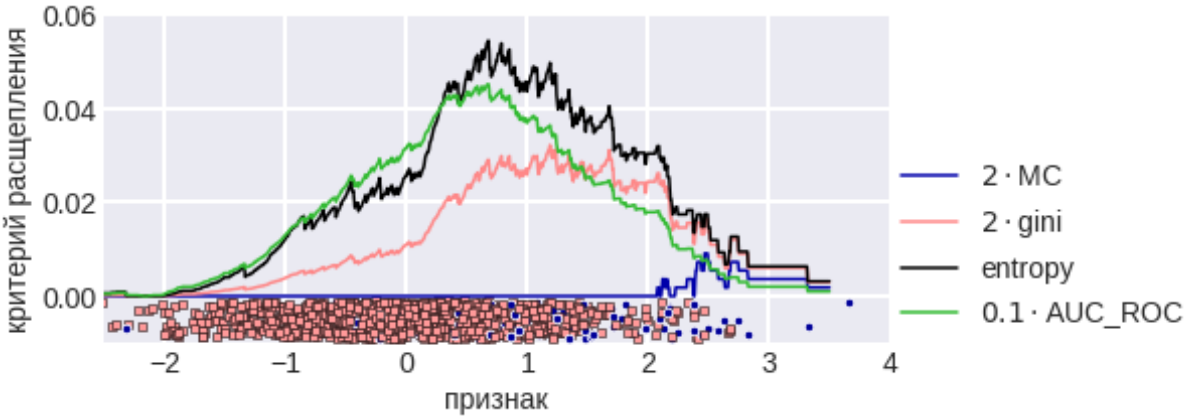

Рис. XX.10. Значения критериев при различных знамениях порога.

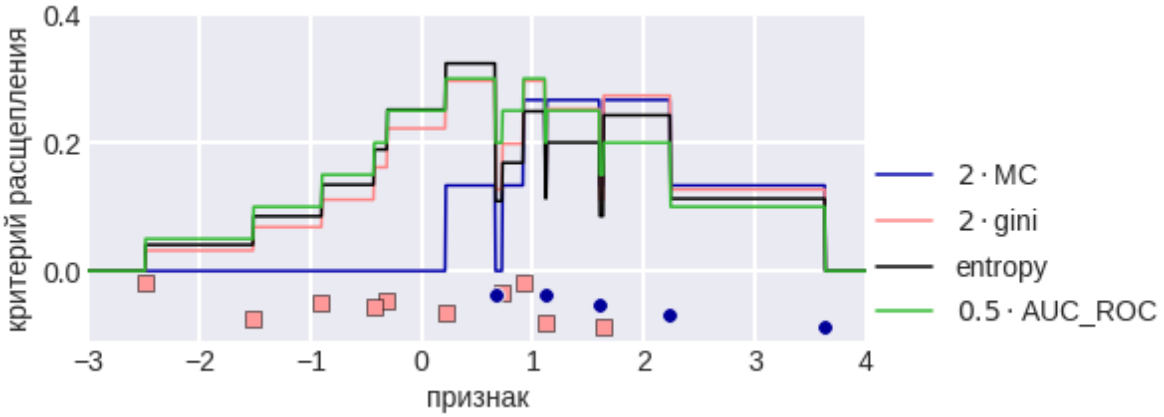

Рис. XX.11. Значения критериев при различных знамениях порога.

Для иллюстрации вычисления подобных функций приведём вычисление $Q(R,1)$ для энтропийного критерия и признака, представленного на рис. XX.11. Порог $\theta = 1$ делит всю область значений признака R, содержащую 15 объектов, из которых 5 принадлежат классу 1, на две подобласти: R_{left} – при значении признака не больше порога, здесь 8 объектов и лишь один из класса 1, R_{right} – при значении признака больше порога, здесь 6 объектов и 4 из класса 1. В результате получаем:

$$\frac{|R_{\text{left}}|}{|R|} = \frac{9}{15}, \frac{|R_{\text{right}}|}{|R|} = \frac{6}{15},$$

$$H(R) = -(5/15)\log(5/15) - (10/15)\log(10/15) \approx 0.918,$$

$$H(R_{\text{left}}) = -(1/9)\log_2(1/9) - (8/9)\log_2(8/9) \approx 0.503,$$

$$H(R_{\text{right}}) = -(4/6)\log_2(4/6) - (2/6)\log_2(2/6) \approx 0.918,$$

$$Q(R,\theta) \approx 0.918 - \frac{9}{15}0.503 - \frac{6}{15}0.918 \approx 0.249 .$$

Это соответствует значению на графике на рис. XX.11.

Формулу для энтропийного критерия можно обосновать теоретически, для этого понадобится условная энтропия, которую мы напомним на примере. Допустим, в некотором мире («в системе») может наблюдаться состояние осадков $X \in \{\text{дождь}, \text{сухо}\}$ – идёт дождь или сухо, а также состояние облачности $Y \in \{\text{облачно}, \text{ясно}\}$ – облачно или ясно, они представлены в следующей таблице (показано сколько в среднем на 10 событий каких пар приходится):

	облачно	ясно
дождь	3	1
сухо	1	5

Можно оценить вероятности пар состояний, тогда энтропия системы – это т.н. **совместная энтропия**

$$H(X,Y) = -\sum_{x\in X}\sum_{y\in Y} p(x,y)\log_2 p(x,y) =$$

$$= -\frac{3}{10}\log_2\frac{3}{10} - \frac{1}{10}\log_2\frac{1}{10} - \frac{1}{10}\log_2\frac{1}{10} - \frac{5}{10}\log_2\frac{5}{10}.$$

Если же мы точно знаем, что сейчас идёт дождь, то это уменьшает неопределённость предсказания состояния, энтропия превращается в

$$H(Y \mid X = x) = -\sum_{x\in X}\sum_{y\in Y} p(y \mid x)\log_2 p(y \mid x) =$$

$$= -\frac{3}{4}\log_2\frac{3}{4} - \frac{1}{4}\log_2\frac{1}{4}.$$

Так появляется ожидаемая условная энтропия, т.е. матожидание значения $H(Y \mid X = x)$ при условии, что знаем X:

$$H(Y \mid X) = \sum_{x\in X} p(x)H(Y \mid X = x).$$

Это неопределённость предсказания состояния при условии, что мы будем знать значение x из X. Тогда изменение энтропии, т.е. насколько знание X уменьшило неопределённость запишется как

$$IG(Y \mid X) = H(Y) - H(Y \mid X)$$

и называется **информационным выигрышем или взаимной информацией (Information Gain / Mutial Information)**. Именно взаимная информация и вычисляется в энтропийном критерии расщепления: см. (XX.4) и данное выражение интерпретируется как изменение неопределённости в предсказании деревом целевого значения при использовании конкретного расщепления (разбиения на две подобласти).

Упомянем также некоторые **тонкости реализации выбора порога расщепления**. Нетрудно заметить, что при выборе оптимального порога

- достаточно рассматривать только «средние точки» (которые находятся между точек выборки на данном признаке), см. рис. XX.12,
- достаточно рассматривать только «границы регионов» (средняя точка должна лежать между представителями разных классов), такие границы соответствуют локальным экстремумам оптимизируемой функции.

Заметим, что границы меняются при спуске по дереву (поэтому их нельзя предпосчитать заранее). Если в каком-то поддереве какой-то признак становится константным, то для него можно не искать расщепление.

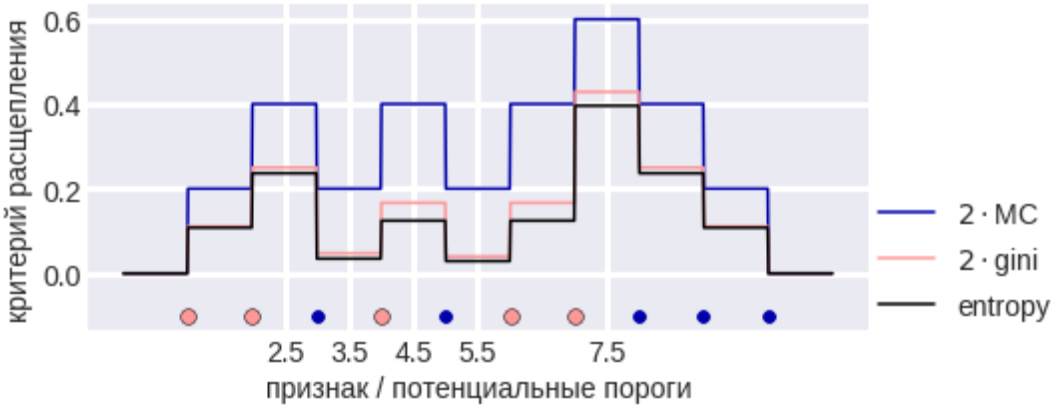

Рис. XX.12. Потенциальные пороги для признака в модельной задаче.

Критерии расщепления в задачах регрессии

В регрессии логика выбора порога такая же, как и при классификации. Здесь под «неоднородностью» естественно понимать дисперсию, поскольку тогда мы будем формировать листы дерева с небольшой дисперсией целевых значений объектов обучения из листа, и ошибка дерева на обучающей выборке также не

будет большой (если приписать листу среднее меток всех объектов попавших в него):

$$H(R) = \text{var}(\{x_i \mid x_i \in R\}).$$

По-прежнему, при выборе расщепления оптимизируем изменение неоднородности (XX.4).

Заметим, что для бинарной случайной величины ($y_i \in \{0,1\}$), если доля объектов с меткой 1 – $p_1 = p$, то доля объектов с меткой 0 – $p_0 = 1 - p$ и дисперсия

$$\text{var}(\{y_i\}) = p(1-p),$$

совпадает с точностью до константы с неоднородностью Джини. Таким образом, мы обосновали неоднородность Джини в задаче бинарной классификации (через связь с регрессией).

Энтропийный критерий связан с условной энтропией, Джини – с дисперсией.

Как долго строить дерево

Критерии останова процесса построения дерева обычно используют следующие:

- ограничение на глубину / на число листьев,
- ограничение на число объектов в листьях[1] / на число объектов, когда делаем деление[2],
- «естественное ограничение»: все объекты одного класса, или его ослабление: почти все объекты одного класса,
- изменение неоднородности (impurity).

В последнем случае, чаще[3] в качестве гиперпараметра метода выступает не порог на изменение неоднородности, а на изменение

$$\frac{|R|}{m}\left(H(R) - \frac{|R_{\text{left}}|}{|R|} H(R_{\text{left}}) - \frac{|R_{\text{right}}|}{|R|} H(R_{\text{right}}) \right) =$$

[1] Например, при таком ограничении равном 5, если в текущей области R 9 объектов, то её невозможно разбить на две подобласти. Если в ней 10 объектов, то возможно лишь одно расщепление: 5 + 5.

[2] Если число объектов в текущей области меньше порога, то расщепление не делается.

[3] Например, в библиотеке sklearn.

$$= \frac{|R|}{m} H(R) - \frac{|R_{\text{left}}|}{m} H(R_{\text{left}}) - \frac{|R_{\text{right}}|}{m} H(R_{\text{right}}). \qquad \text{(XX.5)}$$

Здесь множитель слева увеличивает значения на верхних уровнях дерева (там всё-таки логично делать разбиения, даже если нет удачных, удачные могут появиться впоследствии).

На рис. XX.13-14 показаны деревья при разных ограничениях на максимальную глубину[1], чем больше максимальная глубина, тем лучше дерево настраивается на выборку, но более неестественно выглядит разделяющая / регрессионная поверхность (например, в классификации появляются артефакты в виде узких полос).

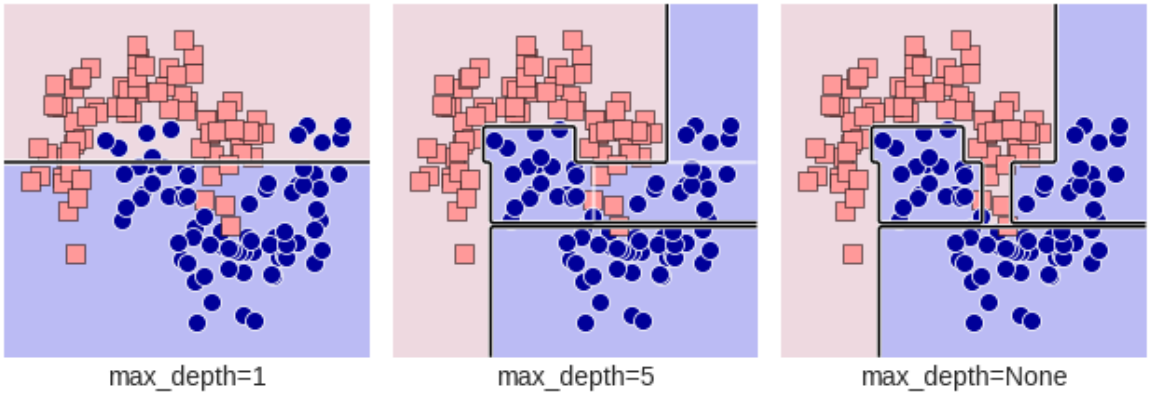

Рис. XX.13. Деревья классификации при разной максимальной глубине.

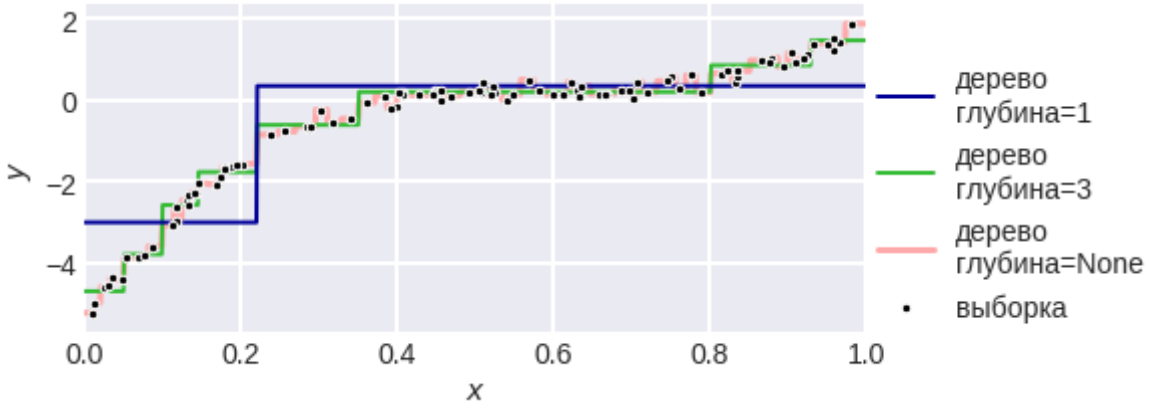

Рис. XX.14. Деревья регрессии при разной максимальной глубине.

Глубокие деревья склонны к переобучению (overfitting), поскольку «затачиваются» на отдельные объекты, см. рис. XX.13-14. Для борьбы с этим используют следующие приёмы:

[1] Выбор max_depth=None соответствует отсутствию ограничения по глубине.

1. Прекращают построение достаточно рано (см. критерии останова), такой приём называют **ранним остановом обучения (stopping early).** В контексте построения деревьев здесь также есть название **пред-подрезка (pre-pruning)**. Можно на отложенной выборке выбрать точку останова.

2. Подрезают уже построенные деревья, используя т.н. **пост-подрезку** (**post-pruning**, см. ниже).

3. Используют деревья в ансамблях (например, в случайном лесе – см. соответствующую главу).

Сейчас подрезка **(**post-pruning) используется крайне редко, только в случаях, если задачу действительно пытаются решить одним деревом (или ансамблем из нескольких). Для подрезки можно использовать отложенный контроль. Тогда в дереве последовательно удаляют нижние ветвления, пока качество на отложенном контроле не уменьшается. На рис. XX.3 видно, что можно удалить ветвление и получить эквивалентное дерево. Можно оптимизировать функционал, в котором штрафуется построение очень глубокого дерева, например, в соответствии с принципом **минимального описания MDL (Minimum Description Length)**

$$\sum_j \sum_{x_i \in R_j} (y_i - a_{R_j})^2 + \alpha |\{R_j\}| \to \min,$$

т.е. к RSS добавляют число областей умноженное на гиперпараметр α (число областей совпадает с числом листьев в дереве). Поскольку каждому листу соответствует метка (значение, которое будет выдаваться алгоритмом на всех объектах, попавших в лист) и это значение является параметром алгоритма, также каждому расщеплению соответствует номер признака и порог – это тоже параметры алгоритма, то таким образом мы штрафуем за число параметров (и стремимся сделать алгоритм проще). Заметим, что значение функционала имеет простую интерпретацию, поскольку расщепление листа потенциально уменьшает RSS (левая часть формулы), но увеличивает число листьев на 1 (т.е. правую часть на α), то значение α – это ограничение на изменение RSS, при котором разумно делать расщепление.

Описанный функционал можно использовать и для пред-подрезки. Оптимальное значение α находят с помощью скользящего контроля, потом с этим значением гиперпараметра дерево перестраивается по всей выборке. Гиперпараметр α регулирует баланс между стремлением обучиться и получить неглубокое дерево.

Важности признаков в дереве

При построении дерева мы производили расщепления, находя признак и порог, которые максимизировали уменьшение неоднородности $Q(R,\theta)$ (XX.4) или нормированной неоднородности $Q(R,\theta)\cdot|R|/m$ (XX.4) (как реализовано в sklearn). Выдвинем гипотезу, что **чем больше признак уменьшает неоднородность, тем он важнее**, тогда важностью признака можно считать сумму уменьшений неоднородностей с помощью этого признака при построении дерева. Иногда считают, что чем чаще признак выбирался в дереве, тем лучше, но здесь легко можно привести контрпример: если в задаче классификации два признака и по одному – бинарному – объекты разных классов почти разделились, то он будет выбран один раз на первом уровне, а далее может всегда выбираться второй признак, поэтому формально расщеплений по нему будет существенно больше. На рис. XX.15 показан похожий пример: достаточно неплохое решение получается деревом глубины 1, которое использует всего один признак, в глубоком же дереве очень много расщеплений было сделано по другому признаку, при этом не кажется, что его значения влияют на целевые.

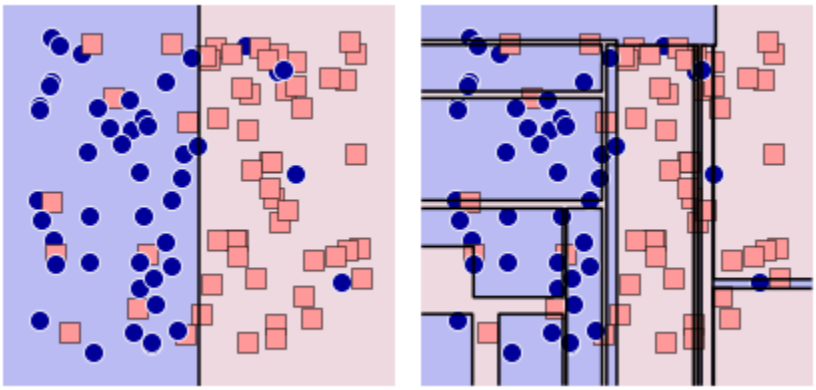

Рис. XX.15. Решение пеньком (слева) и глубоким деревом (справа).

Опишем ещё один способ оценки т.н. **пермутированной важности признака** с помощью дерева, для него нужна отложенная выборка (которая не использовалась при построении дерева). Измеряем качество u на отложенной выборке, затем делаем случайную перестановку значений в j-м столбце матрицы данных отложенной выборки – «перемешиваем значения j-го признака» и измеряем качество u_j на новых данных. Изменение качества[1]

$$\max[u-u_j,0]$$

[1] Естественно, считается что, чем выше качество при «не перемешанном признаке», тем признак важнее.

можно считать важностью j-го признака. Заметим, что при вычислении u_j мы фактически заменяем расщепления, в которых встречается j-й признак на случайные.

Обработка пропусков (Missing Values) с помощью деревьев

В данных могут быть пропущенные (неизвестые) значения признаков, есть много стандартных способов обработки пропущенных значений[1], но при использовании (построении) деревьев можно применять следующие приёмы:

- объекты с пропусками в признаке, по которому происходит расщепление, «проносятся» в обе ветви дерева (не очень выгодно при большом числе пропусков и приводит к потенциально долгой работе алгоритма),
- пропуск соответствует той ветви, которая больше подходит для оптимизации функционала (но тут мы «уравниваем» объекты с пропуском, как будто у них одинаковое значение признака, а оно просто пропущено),
- для неизвестных значений применяют другое «запасное» расщепление, которое иногда называется суррогатным[2].

Поясним на примере последнюю идею **суррогатного расщепления (Surrogate Splits)**. Пусть при построении дерева в некоторой ветке мы дошли до такой подвыборки (это объекты в текущей области)

```
A B C Y=target
0 ? 0 0
1 5 1 0
2 ? 0 0
3 8 1 1
? 7 1 1
? 7 1 1
```

Здесь напрашивается сделать расщепление «A > 2.5», но не понятно, в какое поддерево класть последние два объекта. Нетрудно заметить, что расщепление «B > 6» на объектах с известными значениями признаков A и B совпадает с

[1] см. главу «Предобработка данных».

[2] https://www.learnbymarketing.com/methods/classification-and-regression-decision-trees-explained/

исходным, при этом определено для тех объектов, у которых были неизвестны значения признака A, его и будем использовать в качестве запасного.

Категориальные признаки

Формально при расщеплении мы должны рассмотреть все подмножества множества категорий. В реальности такой перебор сильно сокращают, покажем как на примере бинарной классификации: упорядочиваем категории в категориальном признаке по доле объектов класса 1, после этого каждую категорию заменяем на порядковый номер в указанном упорядочивании, получаем числовой признак, для которого уже находим порог стандартным способом. В расщеплении (XX.2) в качестве множества категорий C выбирается множество категорий, которым соответствуют номера слева от найденного порога.

У указанного способа есть недостатки, например, если категория «маленькая» (мало объектов с таким значением признака), то оценка вероятности класса 1 (которую фактически мы производим, вычисляя долю объектов класса 1) имеет большую дисперсию. Об этом мы поговорим в главе «Предобработка данных».

Также замечено, что если есть категориальные признаки с большим число категорий, то большинство расщеплений производится по ним (что искусственно повышает их важность).

Деревья vs линейные модели

На рис. XX.15 показано сравнение деревьев и линейных моделей на модельных задачах. Деревья разделяют классы довольно неестественно и просто, но в случае сложных нелинейных зависимостей это предпочтительнее использования линейных методов (хотя у них есть довольно мощные обобщения на нелинейный случай[1]).

[1] см. главу «Нелинейные методы».

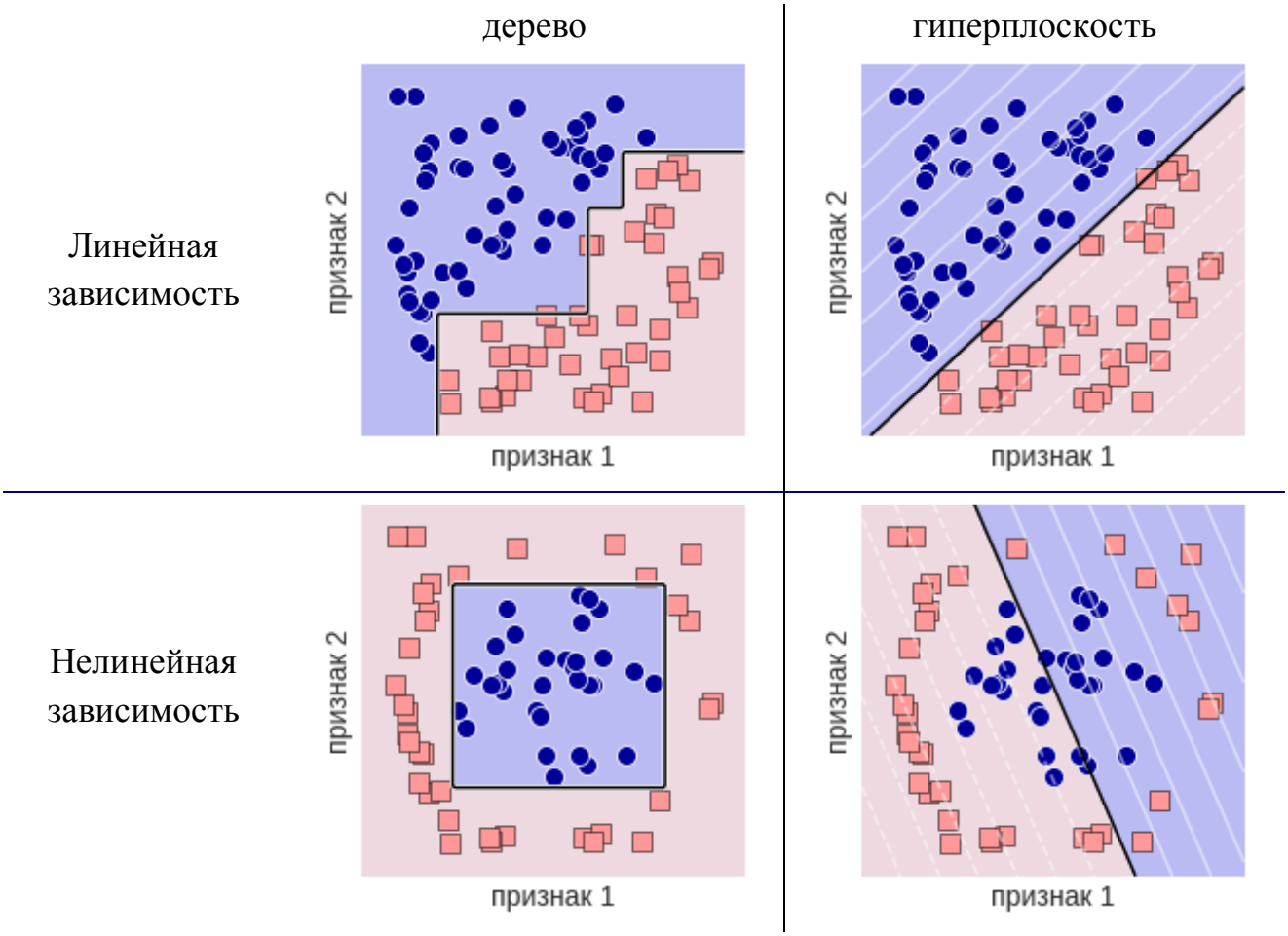

Рис. XX.15. Сравнение линейного классификатора и дерева.

Приложения решающих деревьев

1. Деревья редко используются в машинном обучении как самостоятельные модели. Зато, как мы увидим дальше, очень часто в составе ансамблей. При этом на практике доверяют оценкам важностей признаков, которые получены такими ансамблями.

2. Ещё деревья позволяют генерировать новые признаки. Пусть построено такое дерево:

$$a(x) = \sum_j a_{R_j} I[x \in R_j]$$

Если теперь добавить к базовым признакам признаки вида

$$f_{\text{new}}(x) = I[x \in R_j]$$

(характеристический бинарный признак попадания объекта в область R_j), то линейная модель будет заведомо не хуже, чем построенное дерево, по крайней мере, оно может имитировать дерево – получить такие же ответы, но повышается риск переобучения, если области R_j небольшие. Поэтому такие приёмы применяют для неглубоких деревьев или добавляют характеристические признаки расщеплений.

Методы машинного обучения можно использовать не напрямую, например, для генерации признаков для других методов.

Вопросы и задачи

1. Верно ли, что критерий расщеплений twoing в случае двух классов совпадает с критерием Джини? Обратите внимание, что у этих критериев разная форма записи: через изменение меры зашумлённости и через изменение пропорций классов.

2. Упоминалось, что при построении дерева, если в каком-то поддереве какой-то признак становится константным, то для него можно не искать расщепление. Такое часто бывает в задаче с бинарными признаками. Придумайте для такой задачи эффективную процедуру построения решающего дерева.

3. Мы привели контрпример, который показывает неразумность формализации важности признака как числа расщеплений с ним. Можно ли такие контрпримеры привести для следующих формализаций: средний уровень расщепления, сумма объёмов областей (число обучающих объектов в них), которые расщеплялись с помощью данного признака?

4. Докажите, что описанный способ нахождения расщепления (XX.2) эквивалентен способу, в котором перебираются подмножества множества всех значений категориального признака.

5. Как сделать обучение дерева устойчивым к выбросам (при добавлении в выборку незначительного числа выбросов дерево меняется не сильно)?

6. Получите оценку сложности обучения дерева (как функцию от параметров задачи: число объектов, признаков и т.п.).

Решающие деревья: итоги

Резюмируем плюсы, минусы и особенности решающих деревьев:

возможности

- способны полностью обучиться (выдавать верные ответы) на любой непротиворечивой[1] выборке (при возможности построения неограниченного дерева),
- можно использовать на данных с признаками разных типов (в том числе с пропусками),
- универсальный метод – решение в виде дерева подходит для всех типов задач машинного обучения (для классификации, регрессии, есть деревья для поиска аномалий),
- встроенные отбор признаков (при построении дерева автоматически отбираются признаки, от которых будет зависеть решение),
- нелинейный метод.

качество

- не очень высокое качество решения задачи, большой риск переобучения для глубоких деревьев,
- хороши в ансамблях (подробнее про это поговорим в главе «Ансамблирование»),

эффективность / стабильность

- достаточно быстро строятся, но для больших данных не подходят,
- нет ограничений на распределения признаков,
- «неустойчивый алгоритм» – может существенно измениться при небольшом изменении выборки, сложно применять в задачах с изменяющимися данными.

понимание, интерпретация и анализ

- работу алгоритма просто объяснить неспециалисту,

[1] В выборке нет одинаковых объектов с разными метками.

- ближе к человеческой логики принятия решения, чем другие методы машинного обучения,
- алгоритм можно изобразить (например, в виде картинки на слайде презентации),
- нет красивой аналитической формулы для модели (формула (XX.3) всё-таки достаточно искусственная).

особенности

- не использует геометрию (нет расстояний, метод неметрический),
- устойчив к масштабированию (изменению масштабов признаков),
- устойчив к дубликатам признаков, зависимостям в признаках и т.п.,
- автоматическое решение проблемы пропусков,
- неспособен к экстраполяции,
- может использовать мало признаков (что сказывается на качестве).

Спасибо за внимание к книге!
Замечания по содержанию, замеченные ошибки
и неточности можно написать в телеграм-чате
https://t.me/Dyakonovsbook